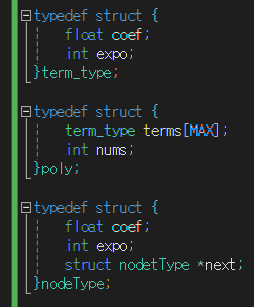
**보고서 #4**

20234014 컴퓨터 공학과 이은정

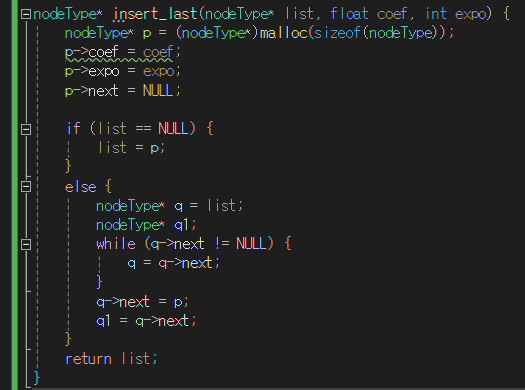
1. 3장에서 배운 다항식의 두 번째 표현 방식으로 2개의 다항식을 초기화하라



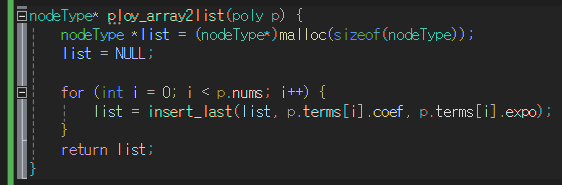
2. 다항식의 한 항을 표현하는 타입 nodetype을 정의하라

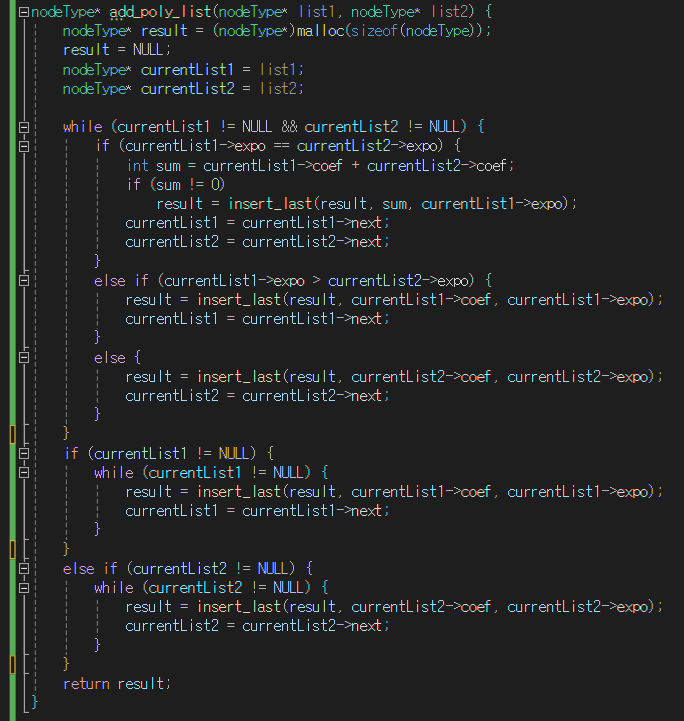


3. list에 새로운 노드 new를 추가하여 반환하는 알고리즘 insert\_last(list, new)를 작성하라.

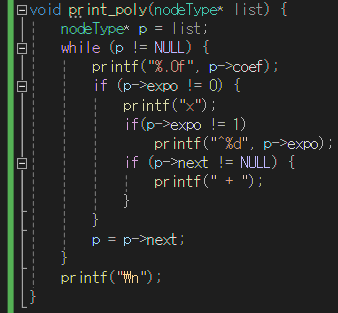


4. 배열로 표현된 다항식을 매개변수로 전달받아서 리스트로 표현된 다항식을 생성하여 반환하는 알고리즘poly\_array2list()를 작성하라.



5. 두개의 다항식을 표현하는 list1, list2를 전달받아서 다항식 덧셈을 수행하고 그 결과 다항식을 생성하여 반환 하는 알고리즘 add\_poly\_list(list1, list2)를 작성하라.

5. 다항식 list를 전달받아서 다음과 같이 출력하는 알고리즘 print\_poly(list)를 작성하라: 10x^3 + 3x^2 – 6x + 1



코드 설명:

#define MAX 100

// 구조체 정의: 다항식 항(term)을 나타내는 구조체

typedef struct {

float coef; // 계수(coefficient)

int expo; // 지수(exponent)

} term\_type;

// 구조체 정의: 다항식(polynomial)을 나타내는 구조체

typedef struct {

term\_type terms[MAX]; // 다항식의 항들을 저장하는 배열

int nums; // 다항식 항의 개수

} poly;

// 연결 리스트 노드 구조체 정의: 다항식 항을 노드로 가지는 연결 리스트

typedef struct nodeType {

float coef; // 계수

int expo; // 지수

struct nodeType\* next; // 다음 노드를 가리키는 포인터

} nodeType;

// 함수 정의: 연결 리스트의 마지막에 노드 추가

nodeType\* insert\_last(nodeType\* list, float coef, int expo) {

// 새로운 노드를 동적 할당하고 초기화

nodeType\* p = (nodeType\*)malloc(sizeof(nodeType));

p->coef = coef;

p->expo = expo;

p->next = NULL;

// 리스트가 비어있으면 새 노드를 리스트의 첫 노드로 설정

if (list == NULL) {

list = p;

}

// 리스트가 비어있지 않으면 리스트의 끝에 새 노드를 추가

else {

nodeType\* q = list;

while (q->next != NULL) {

q = q->next;

}

q->next = p;

}

return list;

}

// 함수 정의: 다항식을 배열에서 연결 리스트로 변환

nodeType\* ploy\_array2list(poly p) {

nodeType\* list = NULL; // 리스트 초기화

// 다항식의 각 항을 연결 리스트에 추가

for (int i = 0; i < p.nums; i++) {

list = insert\_last(list, p.terms[i].coef, p.terms[i].expo);

}

return list;

}

// 함수 정의: 두 다항식을 연결 리스트 형태로 더하기

nodeType\* add\_poly\_list(nodeType\* list1, nodeType\* list2) {

nodeType\* result = NULL; // 결과 리스트 초기화

nodeType\* currentList1 = list1; // 첫 번째 다항식의 현재 노드

nodeType\* currentList2 = list2; // 두 번째 다항식의 현재 노드

// 두 리스트를 모두 순회하며 항들을 더함

while (currentList1 != NULL && currentList2 != NULL) {

if (currentList1->expo == currentList2->expo) {

// 지수가 같으면 계수를 더하고 결과 리스트에 추가

float sum = currentList1->coef + currentList2->coef;

if (sum != 0)

result = insert\_last(result, sum, currentList1->expo);

currentList1 = currentList1->next;

currentList2 = currentList2->next;

}

else if (currentList1->expo > currentList2->expo) {

// 첫 번째 리스트의 항의 지수가 크면 결과 리스트에 추가

result = insert\_last(result, currentList1->coef, currentList1->expo);

currentList1 = currentList1->next;

}

else {

// 두 번째 리스트의 항의 지수가 크면 결과 리스트에 추가

result = insert\_last(result, currentList2->coef, currentList2->expo);

currentList2 = currentList2->next;

}

}

// 남아있는 항들을 결과 리스트에 추가

while (currentList1 != NULL) {

result = insert\_last(result, currentList1->coef, currentList1->expo);

currentList1 = currentList1->next;

}

while (currentList2 != NULL) {

result = insert\_last(result, currentList2->coef, currentList2->expo);

currentList2 = currentList2->next;

}

return result;

}

// 함수 정의: 다항식을 출력

void print\_poly(nodeType\* list) {

nodeType\* p = list;

while (p != NULL) {

// 계수 출력

printf("%.0f", p->coef);

if (p->expo != 0) {

// 지수가 0이 아니면 x 출력

printf("x");

if (p->expo != 1)

// 지수가 1이 아니면 지수 출력

printf("^%d", p->expo);

if (p->next != NULL) {

// 다음 항이 있으면 + 출력

printf(" + ");

}

}

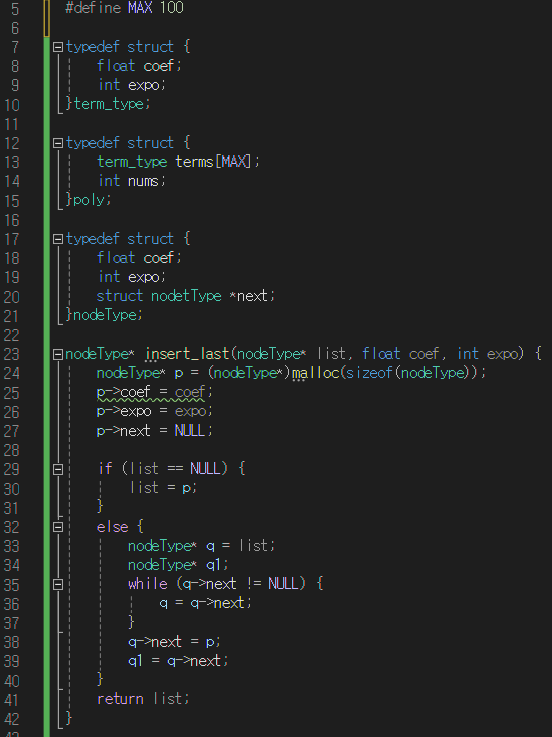
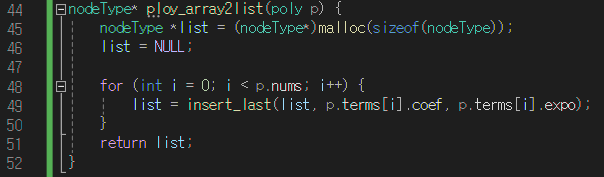
p = p->next;

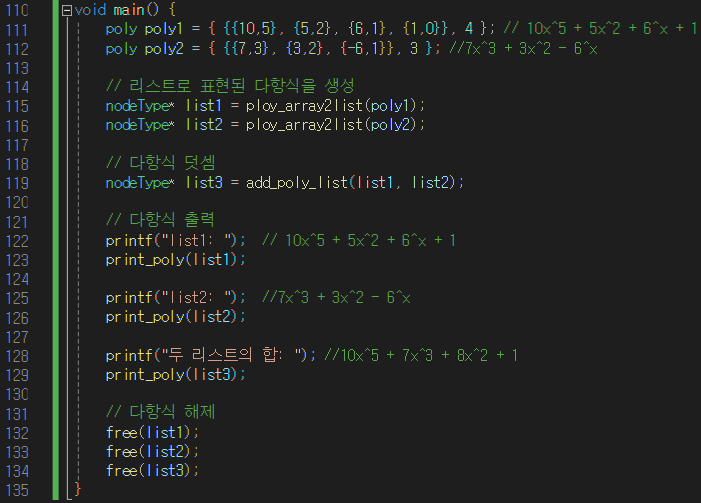
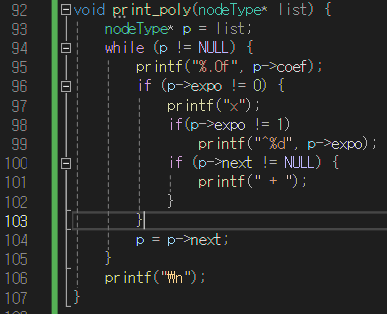
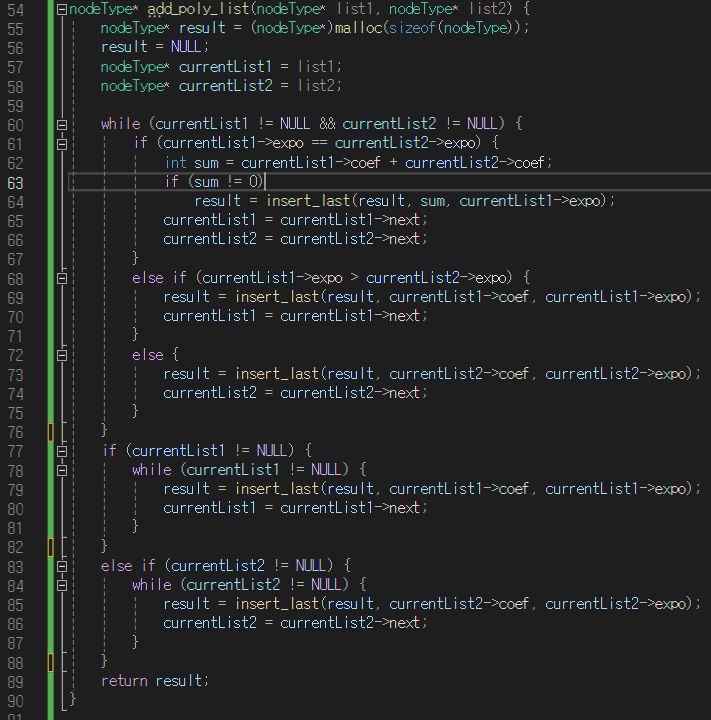
}

printf("\n");

}

C언어 전체 코드:





결과 : 